EE219 Project 2

Clustering

Winter 2018

Xingyi Chen 205032924
Wenfei Lu 505035450

Problem 1

We use function fetch_20 newsgroups to get documents from 8 categories:

class1	class 2
1.comp.graphics	5.rec.autos
2.comp.os.ms-windows.misc	6.rec.motorcycles
3.comp.sys.ibm.pc.hardware	7.rec.sport.baseball
4.comp.sys.mac.hardware	8.rec.sport.hockey

And then transfer it into TF-IDF matrix use function TfidfVectorizer with parameter min_df = 3. **The dimension of TF-IDF matrix is (7882, 27768)**

Problem 2

Apply K-means clustering with k = 2 using the TF-IDF data.

(a) Shape of Contingency Matrix: [[4 3899], [1715 2264]] . As the matrix shows, the documents in class1 are clustered mostly in the cluster 2. However, the documents in class 2 does not show a good cluster result.

(b)

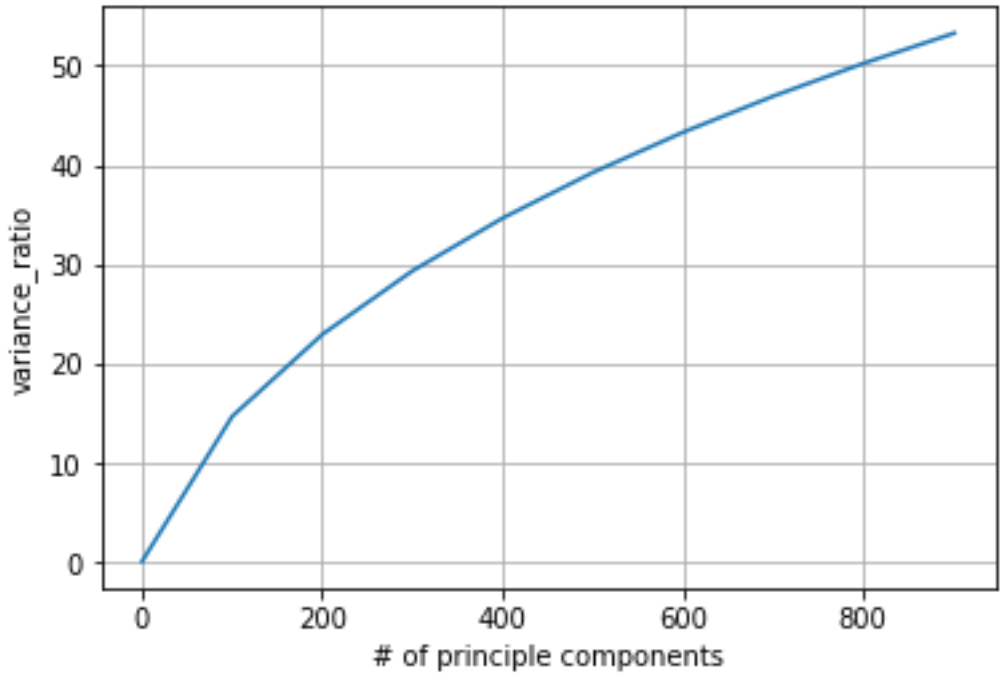
Homogeneity	0.253
Completeness	0.334
V-measure	0.288
Adjusted Rand-Index	0.180
Adjusted mutual information	0.253

According to the 5 measures, we can see that the purely K-means does not have good clustering result.

Problem 3

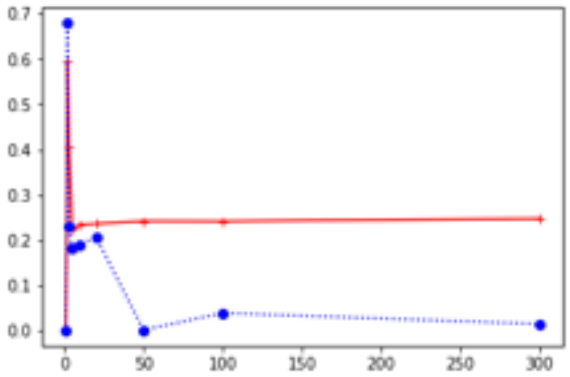
Preprocess the data.

(a) Percentage of variance explained by each of the selected components.

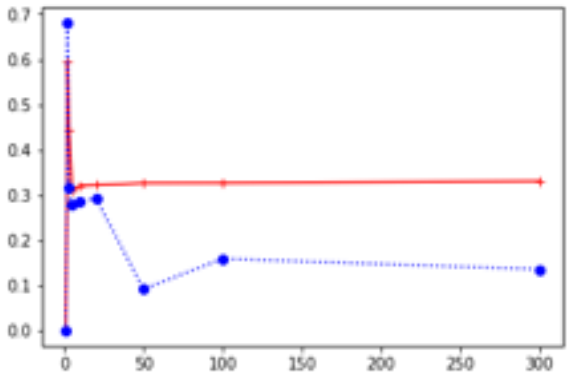


As the plot shows, the more principle components are chosen, the larger precent of variance we get. However, when top 1000 principle components are chosen, the variance ratio is still not 100%. It means that top 1000 principle components could not cover all the information.

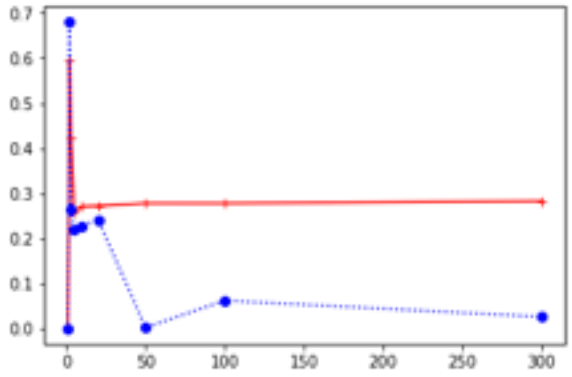
(b) preprocess the data with LSI and NMF
For LSI, we firstly truncate top 300 principle components, and then exclude the least important features to get 1,2,3,5,10,20,50,100 components separately. For NMF, we use nm.fit to reduce the dimension of the data. Following are measure scores v.s. r for both SVD and NMF. The red line is SVD and blue dot line is NMF.



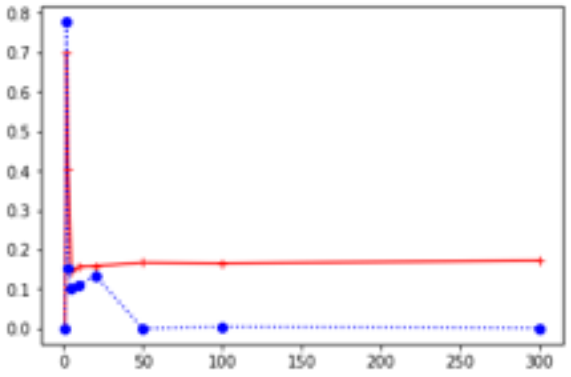
Homogeneity



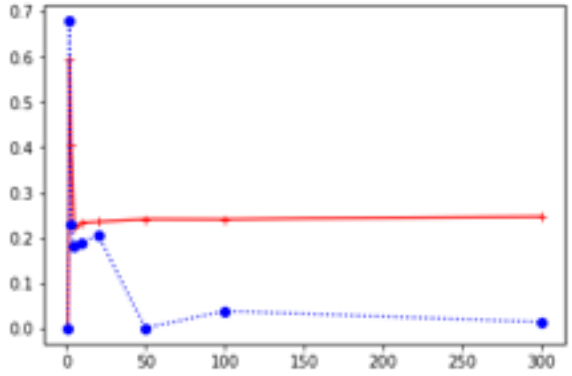
Completeness



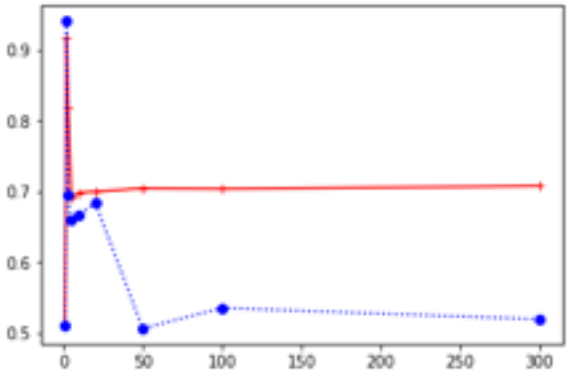
V-measure



Adjusted Rand-Index



Adjusted mutual information



clustering purity metrics

According to the clustering purity metrics (the last picture), r=2 has the best K-means result.
The contingency matrix is as following.

r	LSI	NMF
1	[[1714 2189] [1670 2309]]	[[2200 1703] [2323 1656]]
2	[[3683 220] [429 3550]]	[[3594 309] [158 3821]]
3	[[3866 37] [1401 2578]]	[[4 3899] [1583 2396]]
5	[[3898 5] [2434 1545]]	[[5 3898] [1302 2677]]

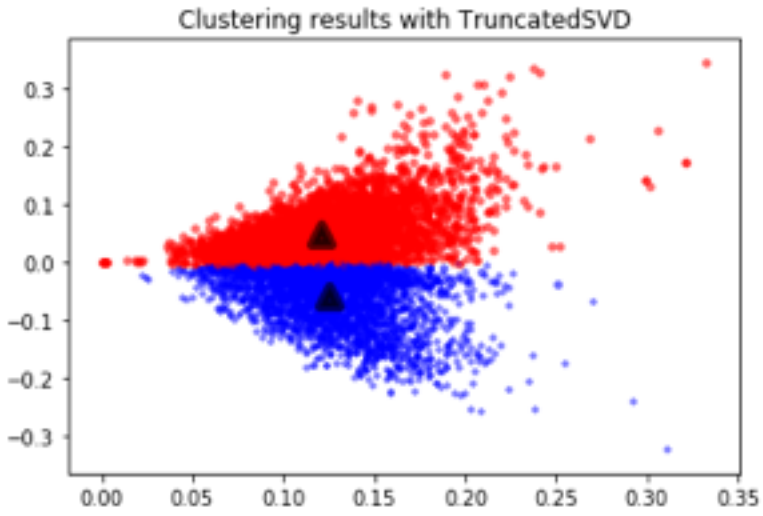
10	[[3 3900] [1603 2376]]	[[3899 4] [2627 1352]]
20	[[3900 3] [2367 1612]]	[[14 3889] [1501 2478]]
50	[[4 3899] [1653 2326]]	[[3893 10] [3979 0]]
100	[[3900 3] [2334 1645]]	[[3901 2] [3663 316]]
300	[[3900 3] [2302 1677]]	[[3790 113] [3979 0]]

From contingency matrix, we could also know that r = 2 has the best result. That's because our raw documents are from 2 classes. So when we divide them into two cluster, we can get a relatively accurate result.

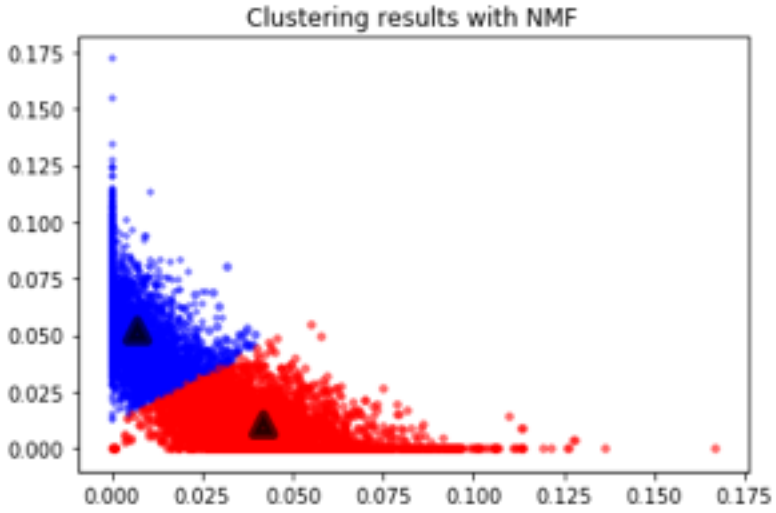
Problem 4

In this problem, to help understand clustering, we visualize the performance and implement with linear and non-linear transformation to it. We found that r=2 works both best for TruncatedSVD and NMF in Problem 3. When applying visualization, we choose r=2 in both dimension reduction function.
In problem a, we visualize the clustering results with TruncatedSVD and NMF separately and compare the performance and difference of the two dimension reduction function.
In problem b, we implemented normalization to both TruncatedSVD and NMF.
In problem c, we implemented logarithm transformation for NMF.
In Problem d, we combined normalization and logarithm with different orders and implemented with NMF.

a) visualization



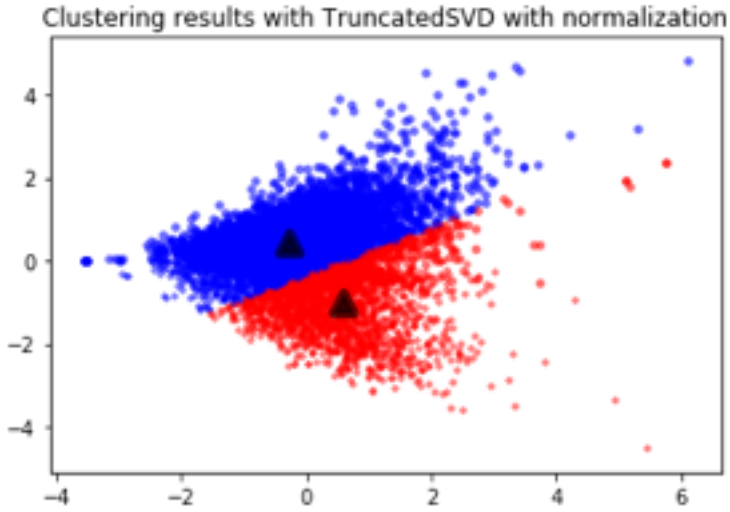
Shape of Contingency Matrix with svd: [[3725 178]
 [526 3453]]
Homogeneity: 0.579
Completeness: 0.581
V-measure: 0.580
Adjusted Rand-Index: 0.675
Adjusted mutual information: 0.579



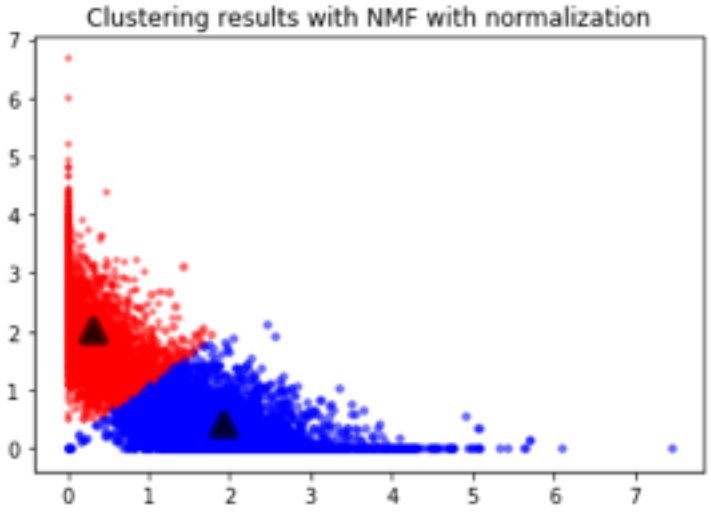
Shape of Contingency Matrix with nmf: [[3589 314]
 [157 3822]]
Homogeneity: 0.677
Completeness: 0.678
V-measure: 0.678
Adjusted Rand-Index: 0.775
Adjusted mutual information: 0.677

Conclusion:
We found that in TruncatedSVD, the data almost separate along the left angle bisector, where the cluster of data points above and below are of different labels. For NMF, the data points mostly distributed beneath the 1/x curve in the plot, while the ground truth labels are approximately separate along the bisector of the 90-degree angle with a considerable amount of false.

b) Normanization



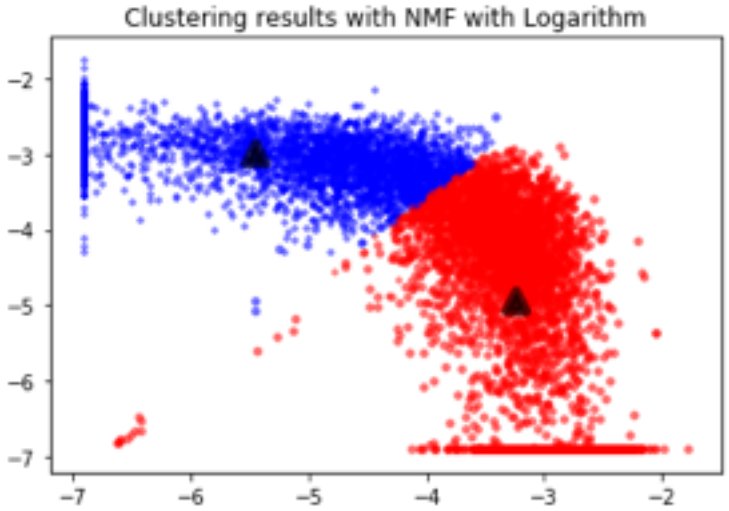
Shape of Contingency Matrix with svd: [[222 3681]
 [2274 1705]]
Homogeneity: 0.247
Completeness: 0.275
V-measure: 0.260
Adjusted Rand-Index: 0.261
Adjusted mutual information: 0.247



Shape of Contingency Matrix with nmf: [[369 3543]
 [3873 106]]
Homogeneity: 0.683
Completeness: 0.686
V-measure: 0.684
Adjusted Rand-Index: 0.773
Adjusted mutual information: 0.683

Conclusion:
For TruncatedSVD, the normalization doesn't perform well while for NMF, the slightly improve the performance.
From the figure of TruncatedSVD without normalization in problem a, we can see that the data points are separated along the x axis. The mean of the blue points are negative and the mean of the red points are positive. By normalizing the data after TruncatedSVD, we lose this characteristic and the mean of these two groups are closer, which decrease the homogeneity and completeness scores.
But for NMF, since all the data points are positive, it does not matter.

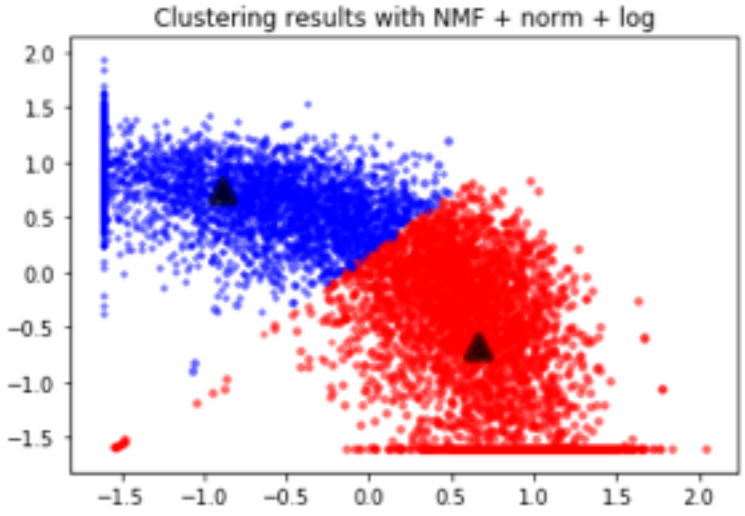
b) Logarithm Transformation



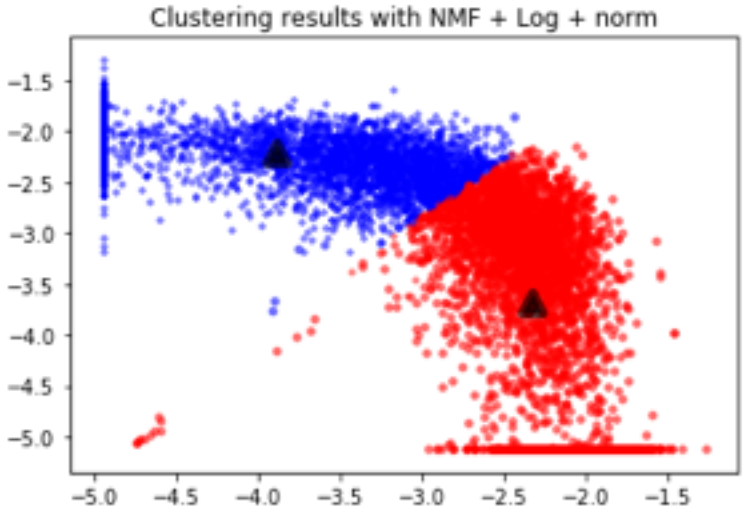
Shape of Contingency Matrix with nmf: [[3681 222]
 [176 3803]]
Homogeneity: 0.712
Completeness: 0.712
V-measure: 0.712
Adjusted Rand-Index: 0.808
Adjusted mutual information: 0.712

Conclusion:
For NMF, the Logarithm Transformation does improve the performance. From previous figure in problem a, we can see that data points are aligned more near the axis and not spread out. The logarithm re-distributes data points and separate them from the axis and hence improve the performance. Especially, we added a small bias to see the improvement and we selected 0.001 here the homogeneity and completeness scores are both improved.

c) Joint normalization and logarithm transformation



Shape of Contingency Matrix with nmf: [[3618 285]
 [127 3852]]
Homogeneity: 0.709
Completeness: 0.710
V-measure: 0.709
Adjusted Rand-Index: 0.802
Adjusted mutual information: 0.709



Shape of Contingency Matrix with nmf: [[3654 249]
 [151 3828]]
Homogeneity: 0.712
Completeness: 0.713
V-measure: 0.712
Adjusted Rand-Index: 0.807

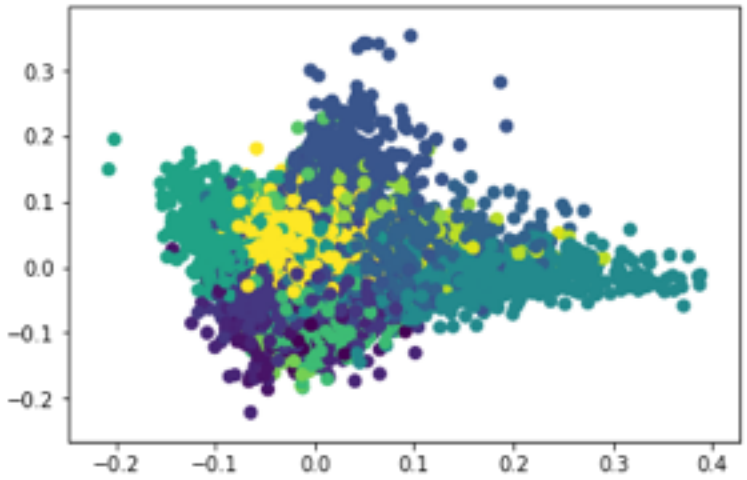
Adjusted mutual information: 0.712

Conclusion:
By combining both transformations with different orders, the performance of clustering remains the same with NMF with only logarithm transformation. Since normalization doesn't change the performance, the result is expectable. The only thing we should take care of is the bias. When implementing logarithm after normalization, the bias is set to 0.1 while implementing normalization after logarithm, the bias is set to 0.001.

Problem 5

1.We get 18846 documents from 20 categories. **The dimension of TF-IDF matrix is (18846, 52295)**
2. We set the number of cluster is 20, which could get better results.

SVD



Homogeneity: 0.292

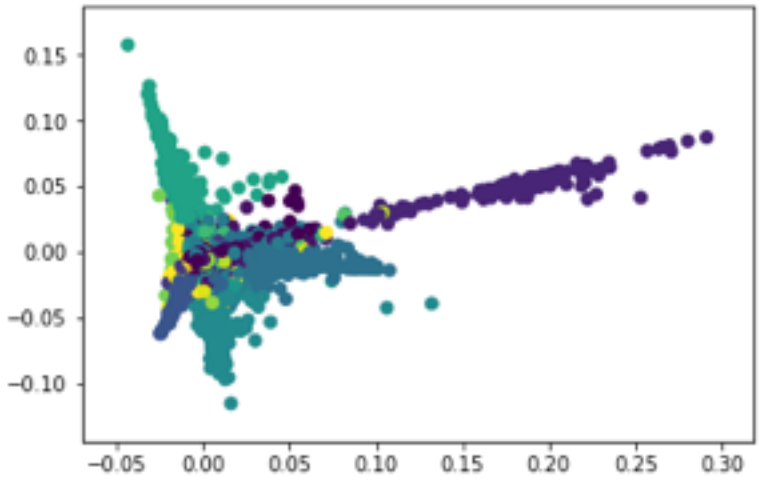
Completeness: 0.384

V-measure: 0.332

Adjusted Rand-Index: 0.104

Adjusted mutual information: 0.290

NMF



Homogeneity: 0.283

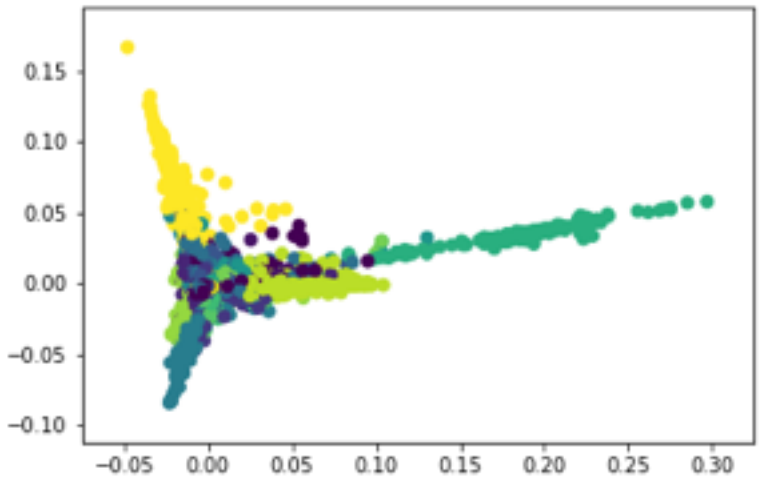
Completeness: 0.373

V-measure: 0.322

Adjusted Rand-Index: 0.094

Adjusted mutual information: 0.281

SVD+norm



Homogeneity: 0.271

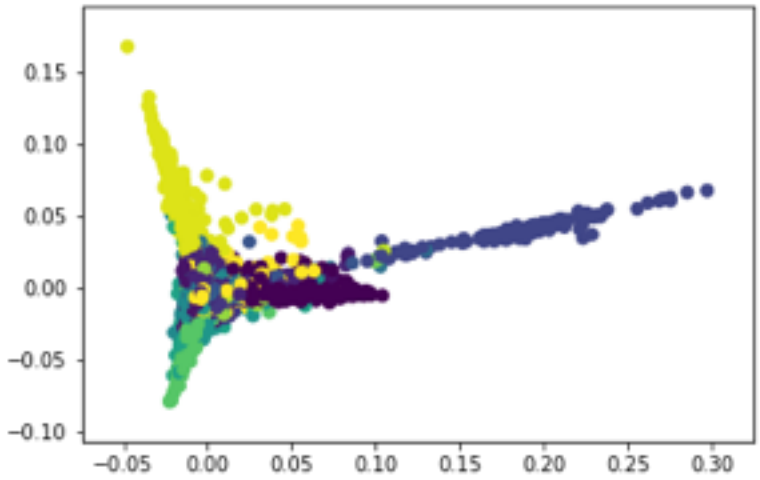
Completeness: 0.340

V-measure: 0.302

Adjusted Rand-Index: 0.096

Adjusted mutual information: 0.269

NMF+norm



Homogeneity: 0.269

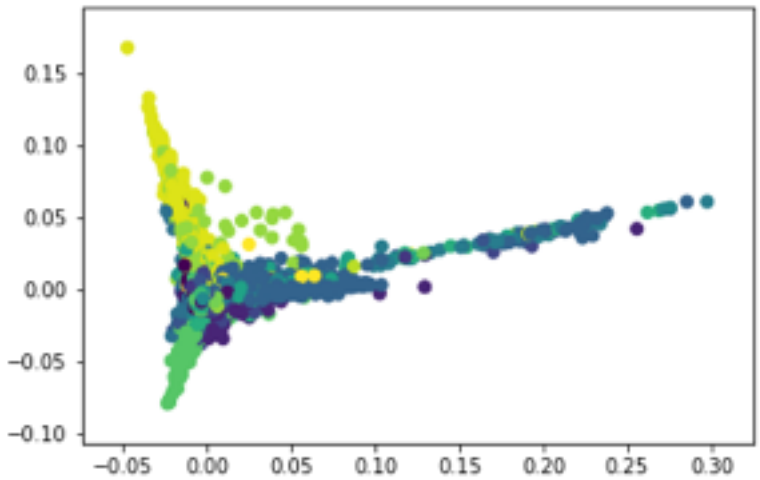
Completeness: 0.357

V-measure: 0.306

Adjusted Rand-Index: 0.089

Adjusted mutual information: 0.266

NMF+log



Homogeneity: 0.405

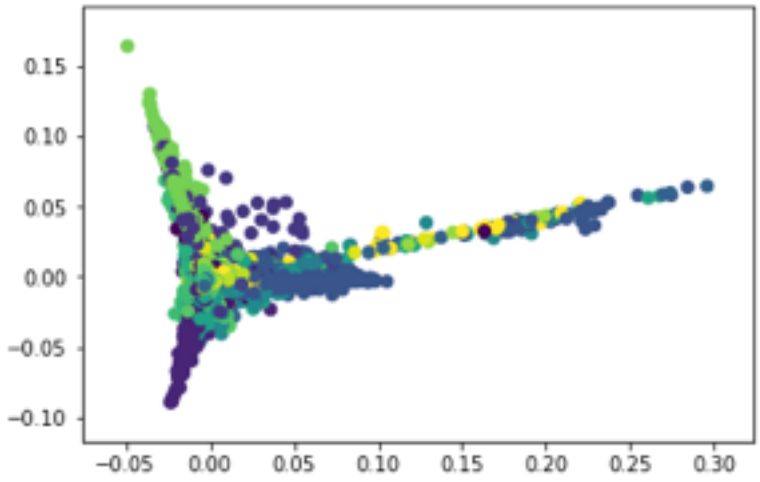
Completeness: 0.416

V-measure: 0.410

Adjusted Rand-Index: 0.251

Adjusted mutual information: 0.403

NMF+norm+log



Homogeneity: 0.385

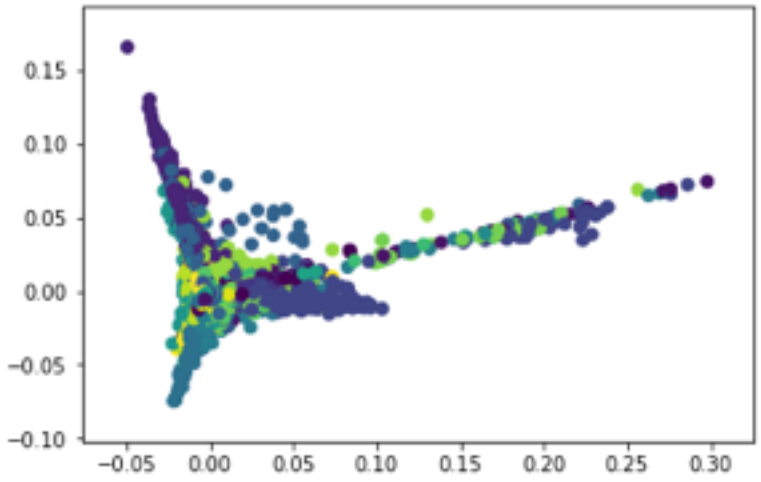
Completeness: 0.390

V-measure: 0.387

Adjusted Rand-Index: 0.236

Adjusted mutual information: 0.383

NMF+log+norm



Homogeneity: 0.384

Completeness: 0.393

V-measure: 0.389

Adjusted Rand-Index: 0.234

Adjusted mutual information: 0.382

Conclusion: AS it shows above, 20 dimension data performs worse than 2 dimension. But with log we can get a better performance of clustering. So it reminds us to transfer data from high dimension to low dimension to get a better result. At the same time, some process like log and normalization will influence the performance of the data processing.